Fundamental concepts of computer science in a Logo-environment

Birgit Wursthorn

University of Education – Institute of mathematics and computer science

71602 Ludwigsburg, Germany, Reuteallee 46

wursthorn@ph-ludwigsburg.de

Abstract

Fundamental concepts of computer science set up the basis of computer literacy because they can be used on a long-term basis and are independent of specific software products. Therefore, we developed a concept to teach these concepts to students aged ten or eleven in mathematics, English, German and music over a whole school year using Logo. In this paper the development of a multimedia story of a family of knights as an interdisciplinary project work in Imagine Logo is described. It is used as an example how fundamental concepts of computer science in connection with the topics of different subjects can be learned and used. Finally the encouraging results of the project work gained from interviews given by the teachers and students are discussed.

Keywords

Syntax, semantics, grammar, functions, object-orientation, programming, multimedia story, project work, Imagine Logo

1. Introduction

The curriculum of the intermediate school in the German state of Baden-Württemberg starts with the statement that computer literacy is essential (Kultusministerium Baden-Württemberg, 2004). Students should be able to evaluate information and communication technology critically and use them in a socially responsible way. An important prerequisite is general computer science knowledge including principles usable on a long-term basis. But the contents of the curriculum don't support such knowledge and the situations in classes are even worse. Students do more or less the same things they already do at home. They write texts and format them, send emails, surf in the Internet and paint pictures. For each task the teachers show the students in given software tools where they can find the required functionality in menus and toolbars. But the underlying concepts of the systems are not explained. Therefore, the knowledge of the students becomes obsolete when software systems are changed.

For this reason it is necessary to start computer literacy education with teaching fundamental computer science concepts as early as possible. At first, beginners should get in contact with the following fundamental ideas of computer science:

- Fundamental principles of computer systems: *algorithm, automation, coding*
- Modelling: *data structure, object-orientation*
- Working techniques: *modularisation, building hierarchies, abstraction, iteration, recursion*

- Standardised description techniques: *flowcharts, syntax diagrams, UML class diagrams, hierarchies, nets*
- Concepts related to a subject: *grammar, syntax, semantics, function*

In form of a spiral the fundamental computer science concepts have to be repeated and intensified in higher classes. If then computer standard programs are used in classes it is also necessary to emphasise the connection between these concepts and the structure and usage of software tools. For example, teachers can point to the structure of menus in word processing systems and show the students the links to the object-oriented architecture of the tools. Another example is a distribution list in an email program which is used to automate the mail process.

2. Teaching fundamental concepts of computer science on the basis of Logo

Excepcially for beginners, Logo as a programming language is suited to support the process of learning the fundamental computer science concepts. Because of the "no threshold" concept of Logo (Harvey, 1982, p.182) learners are able to automate work from the beginning. They start with single commands and after a short while they get bored typing so much. Thus, introducing the `repeat` command is very easy and afterwards used extensively by the students. The same argument can be applied to writing algorithms. When a student draws in his first Logo lesson a square he or she gets in contact with the idea of algorithms.

For developing data structures Logo lists are used. With the help of concrete examples the power of lists to group different things into larger units can be understood. Furthermore, object-orientied concepts can be demonstrated with the turtle viewed as an object. A class `Turtle` has different attributes like `position`, `heading` and `pen`. To change the values of the attributes Logo commands like `forward`, `left` or `penup` can be used and the status of the turtle can be read with functions like `position`, `heading` or `pen`. While working it is important to use the right nomenclature for "class", "object", "attribute" and "method" and to point out the difference between classes and objects.

Because Logo is procedural it is very natural to divide programming problems into small pieces which can be handled more easily. These pieces are packed into single procedures and afterwards used to build up the solution of the whole problem. Consequently, modularisation and building hierarchies are an every day procedure. As mentioned above, the same can be said for iteration. Repeat commands are used permanently. Logo as programming language is recursive. A procedure can be subprocedure of itself. Therefore recursion can be learned with Logo. Most of the time defining new procedures implies abstraction. Simply by writing a square command or a subject function in Logo students can practice this working technique.

Standardised description techniques are paper and pencil methods. Consequently, Logo has in fact no connection to these fundamental concepts. But, for example, it is very simple to realise the descriptions of syntax diagrams in Logo. The rectangles have to be developed as functions and the connections could be established with the Logo function `sentence`. Hierarchies and nets can also be represented with the help of lists in Logo. But they can grow complex very fast and thus not understandable for beginners.

Functions as a concept related to mathematics are permanently used in Logo. In language classes, for example, grammar can be defined with syntax diagrams and implemented in Logo as functions for each part of the sentences. These functions can be used to build sentence generators. The resulting sentences, which are syntactically correct, can be astonishing. They then can highlight the differences between syntax and semantics.

In the following section the course settings of the research project are described, in which the fundamental concepts of computer science were taught. Afterwards, one concrete unit, an interdisciplinary project work with Imagine Logo, is presented and discussed.

3. Course settings

In this research project I developed a concept to teach the fundamental concepts of computer science to students aged ten or eleven. The subjects involved were mathematics, English, German and music. Consequently, teaching these concepts was also meant to coordinate the subjects. Some of the concepts were taught in theory, like the development of UML class diagrams or flowcharts. But if possible the students explored them during the process of problem solving in the context of a subject in Logo micro-worlds.

In each subject the classes took place one hour per week for a whole year. In addition, the students were free to join a study group for two hours where they could work on their own projects. Half of the students accepted this offer. Being a computer scientist, I did all the teaching, but the teachers of the different subjects were present in the classroom.

During the whole school year the students worked only with Logo systems. At the beginning the class used a MSW Logo version in a German translation so that the students could "talk" to the system in their native language. A second reason for using the MSW Logo system was its very easily accessible user interface. After seven months the group changed to Imagine Logo in order to be able to create multimedia presentations and animations, to see the module structure of programs at the user interface and to introduce the concepts of object-orientation nearly without programming.

The school year was divided into three sections.

1. At first the students worked on very small problems to gradually learn the different fundamental concepts of computer science and the programming concepts and syntax of Logo.
2. After eight months they started to work in an interdisciplinary project for five weeks. They created a multimedia story of a family of knights. By this time they used the concepts and programming techniques already learned.
3. At the end of the school year the subject of recursion left and more exercises in modularisation, building hierarchies, modelling and using object-orientation concepts were necessary. Therefore, the students worked without any connection to subject matters. They modified and expanded a car game, developed a painting program and drew and described recursive figures.

So at the end they got to know all the fundamental concepts of computer science.

The results of the research project have shown that even fifth graders when they start to work on small problems at the beginning are able to design large projects after a while. Especially project work in Logo gives students an impression what kind of things computer scientists do. To realise an idea they have to analyse the problem area, design and implement the solution and test the results. Papert describes different levels of learning: "Yes doing is a good way to learn. And it is made better by talking and thinking. But we learn best of all by the special kind of doing that consists of constructing something outside of ourselves" (Papert, 1999, p.13). Consequently, project work meets these requirements best. At the end each pair of students had their product which was very important to most of them in many different ways.

4. The interdisciplinary project of a family of knights

The multimedia story of a family of knights as a project work gave the students the opportunity to

- work on a big project and organise their time table,
- learn subject matter,
- intensify fundamental concepts of computer science,
- create a coherent aesthetic multimedia product and
- prepare a presentation.

The students worked on a project over such a long period of time for the first time. First, the project was well structured in order not to ask too much of the students. In the first lesson a worksheet with the obligatory parts of the project was presented. The students were asked to include in their multimedia product at least

- a story,
- an introduction of people, animals and other things,
- a bird's eye view of a castle, that was axially symmetric,
- a poem generated by the computer,
- a drum rhythm,
- a family tree of their family of knights,
- an interactive riddle for the family tree,
- a title page and
- an ending.

These single parts were described in rectangles on paper because it was expected that the students use them to create a flowchart of their story to plan the arrangement of the different pages in their multimedia presentation. But none of them did. After two weeks they also got a checklist of the obligatory parts of the project to give them guidance. Some students settled it point by point but like before most of them ignored the offer. The story of their family of knights was written as homework and the castles and characters were drawn in art lessons and scanned by the teacher. But all other work was done in the regular lessons. After five weeks the parents were invited for a presentation. Two boys and girls selected by the teacher did a presentation and the others were given the opportunity to show their work on laptops in the classroom.

As subject matters in mathematics, English, German and music axial reflection, the s-genitive, syntax and rhythms were integrated into the project. Each of them was introduced in an extra lesson to explain the subject content and the necessary new functionality of the Imagine Logo system.

4.1. Axial reflection

Before the project work the students had been in contact with axial reflection. At first they were asked to write a command with which the turtle draws the upper right quarter of a castle from a bird's eye view (figure 1).

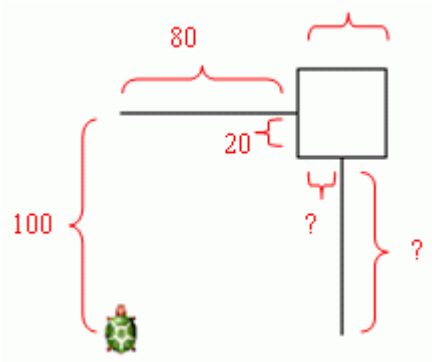


Figure 1. Upper right quarter of the castle

After that they did an axial reflection of the quarter of the castle as an extra exercise with paper and pencil. They learned that three axial reflections are necessary to get the whole castle. Depending on the heading of the turtle they described the reflection axes as straight lines

- along the heading and
- perpendicular to the heading.

Furthermore they distinguished the upper left and the lower right quarter, which were performed as a reflection on a single axis and the lower left quarter which is the result of a concatenation of two reflections. Analogically the following three new commands were introduced:

```
reflect_along_heading
reflect_perpendicular_heading
reflect_along_and_perpendicular_heading
```

The commands need as an input the name of another command in which a figure is drawn. They then draw the reflected figure on screen. The reflection axes are dependent on the position and heading of the turtle at the end of the drawing process of the original figure. Therefore, the students were obliged to write commands which do not change the state of the turtle. Some of them stopped the drawing process at the second straight line. The result of using this command with `reflect_along_heading` is shown in figure 2.

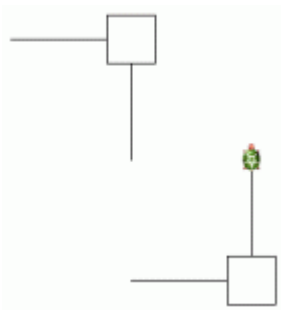


Figure 2. Reflection with a module changing the state of the turtle

A very important result in the area of the fundamental computer science concepts has been the insight that working in a modularised system requires not only knowledge of the interfaces of modules. It is also necessary to know and follow the specifics of modules so that you do not get unexpected results. Therefore, it would be better to develop procedures in a general way.

4.2. English s-genitive

As a topic in English the s-genitive was integrated in an interactive riddle asking for blood relationship. At first a family tree of a knight's family was developed on an extra page. After that the students were asked to think of six questions using the s-genitive like "Is Alfred knight Kunibert's father?". Some of the answers should be true and others false. The last step was the implementation in Imagine Logo as an interactive riddle with yes- and no-buttons, colour and text indicators for the correctness of answers and a button to restart the riddle. An extract of the results of two boys is shown in figure 3.

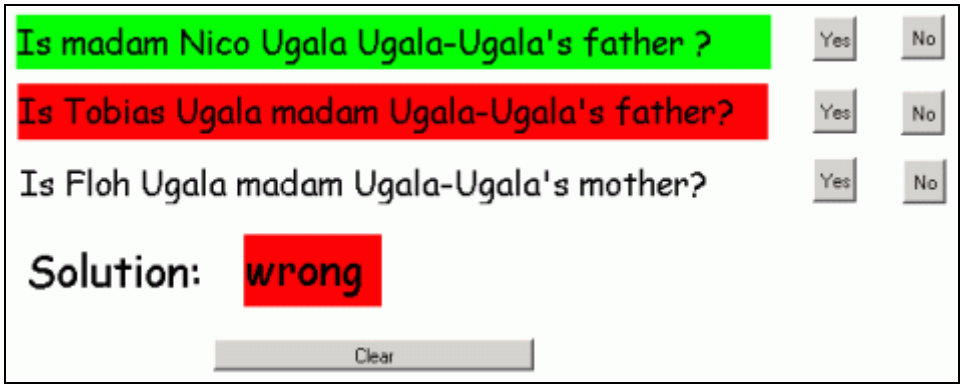


Figure 3. Interactive riddle to a family tree

At the end the page contained eight text boxes and thirteen buttons altogether. Because the students were not used to working with such a lot of objects they were explicitly asked to give the text boxes and buttons meaningful names. Furthermore it was suggested to use a meaningful and consistent nomenclature like `text_question1`, `button_question4_yes` or `button_clear`. To colour the background of the text boxes in green or red and to set the corresponding text in the global solution text box, the two new attributes "background colour" and "text value" of the class `TextBox` were introduced. In addition, the methods to change the attribute values were presented.

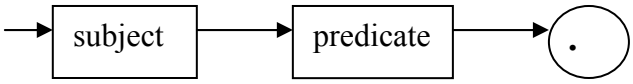
In this part of the project the focus was on a deeper understanding of object-oriented concepts. At the beginning of the project work the students already knew methods to change attributes of objects with the help of the context menu or user interface. They used these methods intuitively. They had also noticed differences in the user interfaces of different object types. While working on the interactive riddle they gained a more accurate picture of object-oriented concepts because they selected the objects not directly with the mouse but in a more abstract way by naming them. The same can be said for changing attribute values. Here instruction statements like

```
text_question1'setBGColour  "red
```

look like native language statements and make the structure of the object selection, the linking to a method and the specification of an attribute value clear.

4.3. Syntax

In German, the students made the computer generate a poem. A love poem of the knight to his mistress was intended but the subject turned out to be too delicate. Thus, it was changed to things the knight and possibly a dragon or another figure were doing. At first the teacher together with the class developed the structure of the verses of the poem generated. An introductory example combined of a subject, a person, and a predicate was given. Then the syntax of the verse



was visualised in a syntax diagram on the black board and matching examples were built. After that the students were asked to do it on their own and implement for each rectangle of the syntax diagram its own function. Any function should randomly pick an element from a defined list with alternative examples. The students had to develop a user interface on which the poem could be created by a reader. So they implemented a button and defined in its push event the building of a verse with the Logo function `sentence` and the newly defined functions. The tests showed the students that pressing the button does not create a poem because each new verse was written in the same position so the result was unreadable. Knowing how to move the writing turtle two possible solutions were built. One group changed their verse generator and moved the turtle directly after writing the verse on the screen. Others implemented an extra button to create a new line. One group was more innovative. Instead of creating new lines they rotated the turtle so that a poem like a sun was formed.

Figure 4. Computer generated verses as sun-rays

Another button was added to clear the whole poem. While developing und using the verse generator the computer science concepts of grammar, syntax, semantics and syntax diagrams were repeated.

4.4. Rhythm

In music the time signature was in the focus. The Logo micro-world gave the students the option of a high and low timbale sound. Furthermore they had the opportunity to choose between semibreves, minims, crotchets and quavers. Each different note could be selected by pressing a button on the user interface. The note was then drawn on the screen. To structure the theme there was also a button to draw a bar line but no algorithm checked whether the selected notes between two bar lines were adequate for the given time signature. Figure 5 shows an example of a rhythm.

Figure 5. Visual representation of a rhythm

The page with the user interface to compose rhythms also contained a button to play them. But to use the rhythms on other pages they had to be saved with the command `save_rhythm` and replayed with `play_rhythm`.

4.5. Animations, multimedia and navigation

Many students also integrated animations of objects like knights, guards of the castles or dragons in their project. They often followed a straight path at different rates or walked around randomly. In addition some students recorded dialogues or noises of specific situations like a wedding song or cries for help. They were integrated as a multimedia object into the project. Others used the melody micro-world of Imagine Logo to compose melodies. The understanding of the story was supported by these animations and multimedia elements. Because it is not possible to change the order of pages in Imagine Logo in the toolbar and because the story was not designed consecutively it was necessary for the students to implement navigation throughout their story. Most of them put two arrow buttons on each page to make a linear navigation available. But some students also implemented branches to allow the user things like looking up the solution of the riddle in the family tree.

5. Results

Although the project consisted of many conceptually different tasks and parts, the project results of the young students were remarkable. At the end of the school year the teachers characterised the results of the project in interviews as impressive, amazing and "gigantic". They said that the students had learned a lot and that each child had had the possibility of profiting from his abilities during the project. They valuated the results as a great interdisciplinary achievement for all of them. They also described the behaviour of the students as motivated and euphoric. They saw the students have fun while working and felt the strong identification of the students with their projects.

Six students were also selected to do an interview. They emphasised in their interviews that they liked the pictures, melodies and rhymes. More general they enjoyed not having to fill in worksheets and more freedom in class. One girl mentioned a problem with her partner and for two of them it was hard to type the story. The time period of five weeks was long enough. All persons interviewed except for one student saw the time given as adequate. Asking the students for further elements they would have liked to implement, only the good students mentioned animation, voices and pictures. From the point of learning subject matters and fundamental concepts of computer science more time would have had no advantage.

Looking at the parts of the project and not at the result, it can be stated that not all tasks were performed in the required way. In mathematics, for example, the students were asked to start with the quarter of the castle in figure 1 to gain an impression of its appearence from an bird's eye view. Afterwards they were asked to experiment with the geometric figures of the tower and the wall to design a castle with a more interesting ground plan. But all of them were satisfied with their first solution. They mentioned laziness, difficulty in designing other forms and their time budget which they prefered to invest in other parts of their work. Others were satisfied with an easy solution or only felt up to do those things the teachers wanted them to do. The teachers also said, that the students preferred to use other alternatives and avoided additional labour. Furthermore, two of the teachers mentioned that students this young could not picture different types of castles and therefore they hadn't made an effort.

Playing with words and sentences like a linguist was also no success in the context of the project. The students didn't collect a lot of alternatives for the different parts of the verses.

The Logo lists often contained only two or three variations. Even a girl which stated in the interview that the work with the rhymes was fun for her didn't create a larger solution. On the other hand the riddle can be valued as a great success. While developing the family tree with different colours, pictures and tree structures many students enjoyed doing creative and artistic work. Fascinated by the idea of creating an interesting product they didn't avoid the hard work of constructing the user interface of the interactive riddle although they had to handle many objects and write object-oriented code. The micro-world to create the drum rhythms only was used as a toy. The students didn't heed of pitches and rhythm schemes.

Furthermore after working for a long time on small problems the students didn't have any difficulty in using fundamental concepts of computer science on their own in order to perform their work in the larger project.

6. Conclusion

The results of the entire research project have been very encouraging because it could be shown that fundamental concepts of computer science can be taught within another school subject like mathematics, English, German and music and even fifth grader are able to learn them. Furthermore some fundamental concepts of computer science support the learning of subject matters and sometimes they even connect different subjects. In addition to that the multimedia project of the family of knights presented in this paper gave the students the opportunity

- to plan and organise their work in a self-directed manner over five weeks,
- to learn content of the curriculum of different subjects while creating one product,
- to apply the fundamental concepts of computer science,
- to engage in a meaningful important work and
- to use information technology.

Thereby their computer literacy competencies grew. For that reason it would be reasonable to perform a similar project at each class level and prepare the implementation of them with short teaching units to consolidate the fundamental concepts of computer science already learned and to introduce new ones. Consequently, new ideas of interdisciplinary, meaningful and adequate projects for different class levels have to be developed. For successful projects it will be important that the different requirements like learning content of the curriculum, getting in contact with concepts of computer science or considering the interests of the students don't lead to far-fetched topics.

Reference

Kultusministerium Baden-Württemberg (2004), *Bildungsplan für die Realschule*, Neckar-Verlag, Villingen-Schwenningen.

Harvey B (1982), *Why Logo?*, Byte, Vol. 7 No. 8, 1982, Mc Graw Hill Publication.

Papert S (1999), *What is Logo? Who Needs It,* Logo Philosophy and Implementation, Logo Compuer Systems Inc.

Logotron (2001), *Imagine Logo*, http://www.logo.com/imagine (2005, May 03).